EU Framework Program for Research and Innovation (SC5-18a-2014 - H2020)

Project Nr: 641538

Coordinating an Observation Network of Networks EnCompassing saTellite and IN-situ to fill the Gaps in European Observations

Deliverable D4.2

Observation inventory description and results report

Version 1

Due date of deliverable: 31/12/2015
Actual submission date: 12/02/2016

Document control page				
Title	D4.2 Observation inventory description and results report			
Creator	CNR			
Editors	CNR			
Description	Report on the observation inventory description (process, architecture and APIs) and results report.			
Publisher	ConnectinGEO Consortium			
Contributors	ConnectinGEO Partners			
Type	Text			
Format	MS-Word			
Language	EN-GB			
Creation date	15/01/2016			
Version number	1			
Version date	25/01/2016i			
Last modified by				
Rights	Copyright © 2015, ConnectinGEO Consortium			
Dissemination level		CO (confidential, only for members of the consortium)		
	X	PU (public)		
		PP (restricted to other programme participants)		
		RE (restricted to a group specified by the consortium)		
	When restricted, access granted to:			
Nature	X	R (report)		
		P (prototype)		
		D (demonstrator)		
		O (other)		
Review status		Draft	*Where applicable:*	
	X	WP leader accepted		Accepted by the PTB
		PMB quality controlled		Accepted by the PTB as public document
	X	Coordinator accepted		
Action requested		to be revised by all ConnectinGEO partners		
		for approval of the WP leader		
		for approval of the PMB		

		for approval of the Project Coordinator
		for approval of the PTB
Requested deadline		

Revision history			
Version	Date	Modified by	Comments
0.1	29/01/2016	CNR_MS, CNR_SN	First draft.
0.5	10/02/2016	CREAF_JM 52N_SJ	Comments/contributions to the draft
1.0	12/02/2016	CNR_MS, CNR_SN	Final version integrating partners' comments and contributions.

Contributors	
Acronym	Full name
CNR_MS	Mattia Santoro (CNR)
CNR_SN	Stefano Nativi (CNR)
CREAF_JM	Joan Maso (CREAF)
52N_SJ	Simon Jirka (52N)

Table of Contents

Executive Summary

The ConnectinGEO Observation Inventory (OI) is created and populated using the current information in the metadata concentrated in the GEO Discovery and Access Broker (DAB) of the GEOSS Common Infrastructure (GCI) to analyse the observations and measurements currently available in it.

WP4 defined a high-level process for the population of the Observation Inventory: (i) retrieve the full metadata content for each record in the GEO DAB, (ii) extract/Infer extra semantics (connecting to external knowledge systems when needed), and (iii) generate enriched metadata and write it to the OI.

The OI system architecture was designed and developed. The first version of the OI was created and populated using the current information in the metadata concentrated in the GEO DAB. The first population process was run in December 2015, resulting in a total of more than 1.6M harvested metadata records.

The developed OI is accessible online and can be used as a data source by different analysis tools, which create plots, reports, or summary statistics useful for the ConnectinGEO gap analysis.

A simple Web Client was developed to demonstrate how to interrogate the OI and provide also basic examples of how the developed OI can be used by web-based analysis tools.

The developed framework is ready to be extended and complements current information with data extracted from additional external sources (URR 2.0, scientific literature DBs, etc.) and the results of ConnectinGEO Task 2.3.

1 Introduction

ConnectinGEO's primary goal is to link existing coordinated Earth Observation networks with science and technology (S&T) communities, the industry sector and the GEOSS and Copernicus stakeholders. An expected outcome of the project is a prioritized list of critical gaps within the European Union in observations and the models that translate observations into practice relevant knowledge.

The project defines and utilizes a formalized methodology to create a set of observation requirements that will be related to information on available observations to identify key gaps.

The gaps in the information provided by current observation systems as well as the gaps in the systems themselves are derived from five different threads. One of these threads consists in the analysis of the observations and measurements that are currently registered in GEO Discovery and Access Broker (DAB). To this aim, an Observation Inventory (OI) is created and populated using the current information in the metadata concentrated in the DAB.

1.1 Scope & purpose of the document

This document describes the process defined to populate the ConnectinGEO Observation Inventory and the resulting system architecture taking into account the requirements and database schema from ConnectinGEO Deliverable 4.1. Finally, this document provides information on how to systematically access the created OI for performing the gap analysis.

2 Observation Inventory Population

The high-level process to populate the ConnectinGEO Observation Inventory (OI), depicted in Figure 1, can be split into three steps:

1. Retrieving the full metadata content for each record in GEOSS DB;
2. Extract/Infer extra semantics, connecting to external knowledge systems when needed;
3. Generate enriched metadata and write it to the Extended OI DB.

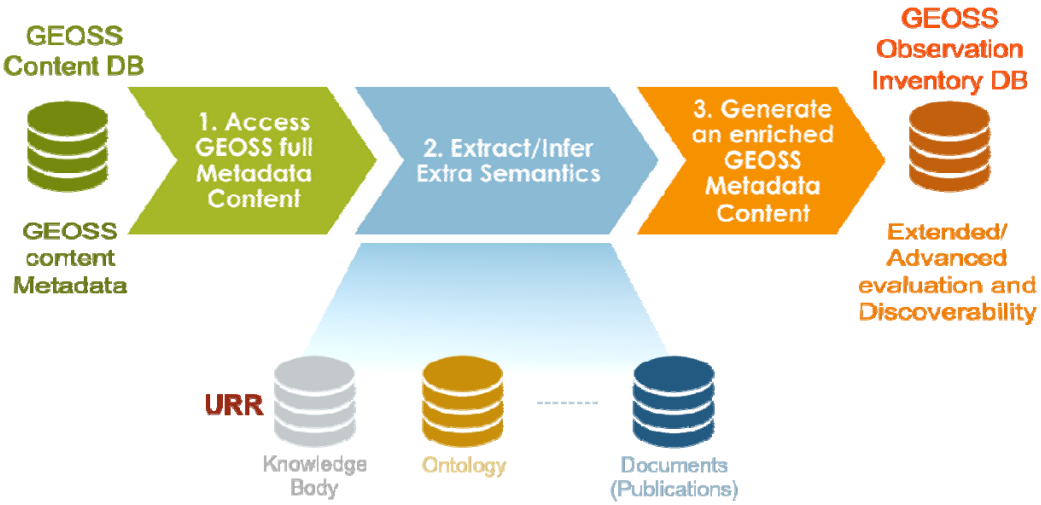

Figure 1 - Populating OI: High-Level Process

The following sections provide more details about the above steps.

2.1 Getting Full Metadata Content

GEOSS metadata content is stored in a No-SQL DB. The content of this DB is generated by the GEO Discovery and Access Broker (DAB) when GEOSS Supply Systems are harvested.

Retrieval of metadata is executed by the GEO DAB components which read/write content during harvesting phase. This way, full metadata content (including GEO DAB specific fields) is retrieved and passed to the next step.

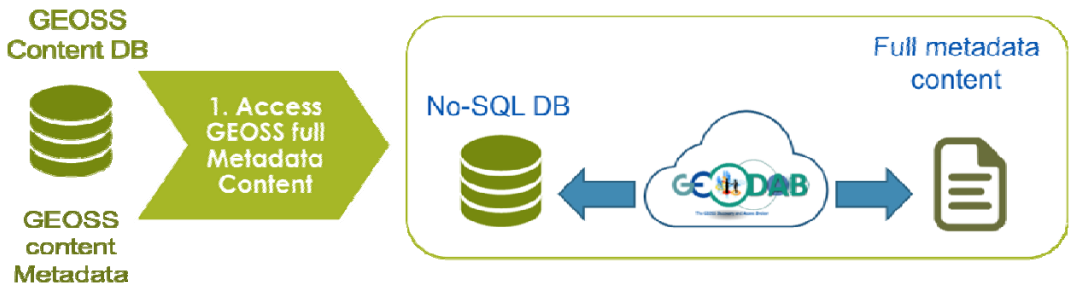

Figure 2 – GEOSS Content Metadata Retrieval

2.2 Extract/Infer Extra Semantics

After retrieving the full metadata, this must be enriched by extracting/inferring additional information. This is achieved by applying the well-known *Enricher* design pattern[1]. In a nutshell, this pattern is used when transferring content from one system (origin system) to another one (target system) and the target system requires more information than the origin system provides. The pattern design adds a new component called *Enricher*. This new component is in charge of retrieving additional information from external resources, exploiting original content information if needed (e.g. identifiers, spatial coverage, etc.). Finally, the Enricher produces the enriched output message that is sent to the target system.

Figure 3 depicts the application of the Enricher pattern to the Observation Inventory use case. The origin system is the GEOSS Content DB, thus the input of the Enricher is the full metadata content retrieved in previous step. The target system is the OI DB, thus the output of the Enricher is an enriched version of the metadata content complying with the OI DB schema. The external resources needed to enrich the content are the ones identified in ConnectinGEO D4.1, for simplicity only URR is represented in figure. These external resources are accessed by the GEO DAB, extending its functionalities where needed. Finally, a set of rules is also present in the depicted schema. These rules define the business logic implemented by the Enricher to infer new information.

It is worth to notice that in Figure 3 depicts more than one Enricher. This is because a set of dedicated Enrichers is envisioned, each one dedicated to provide additional information of specific type (e.g. accessibility, data type, measurement, etc.).

[1] http://www.enterpriseintegrationpatterns.com/patterns/messaging/DataEnricher.html

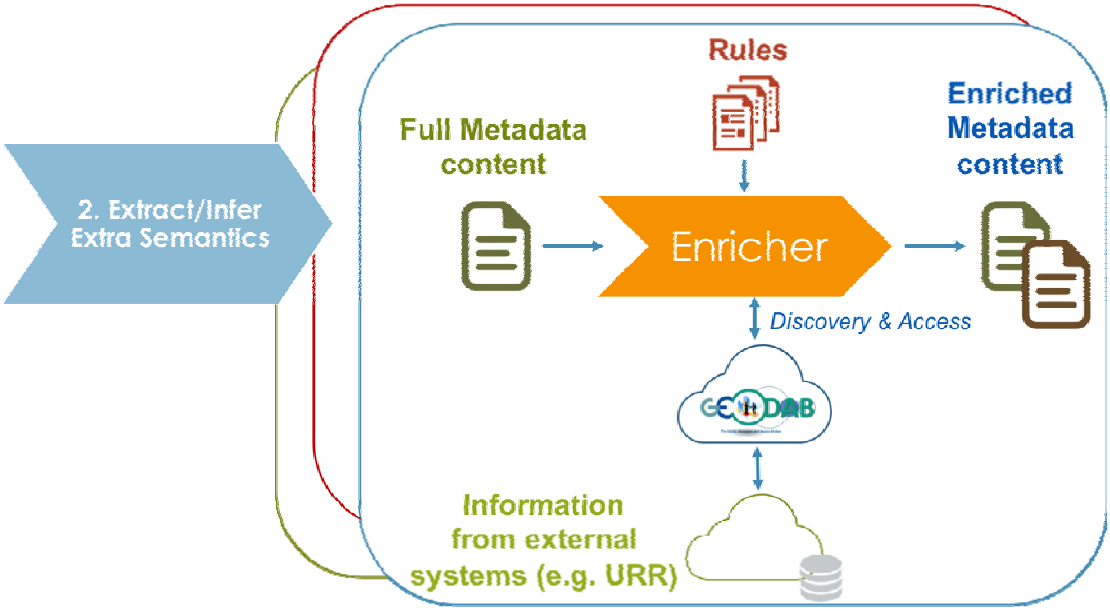

Figure 3 – Enriching GEOSS Metadata Content

2.3 Generate Enriched Metadata Content

After Enrichers have produced the enriched metadata content, this is written to OI No-SQL DB by the GEO DAB. To do this, GEO DAB functionalities are extended to support the new queryable fields required by the OI DB.

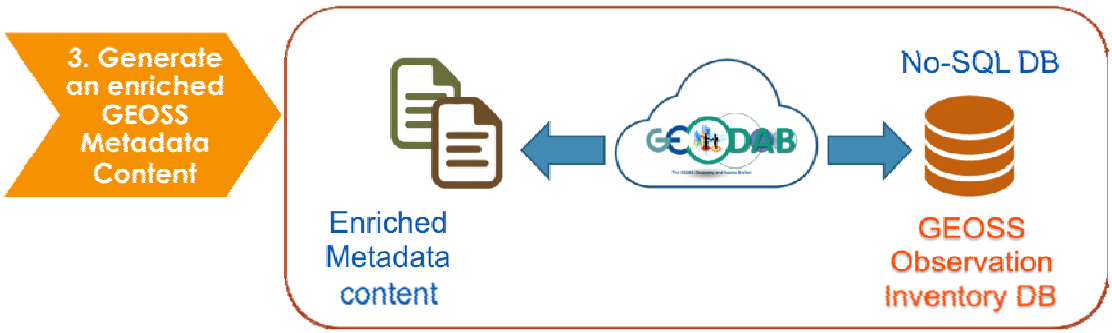

Figure 4 – Storing the Enriched Metadata Content

3 Observation Inventory Architecture

To implement the process described in section 2, the ConnectinGEO OI high-level component architecture depicted in Figure 5 was designed and developed.

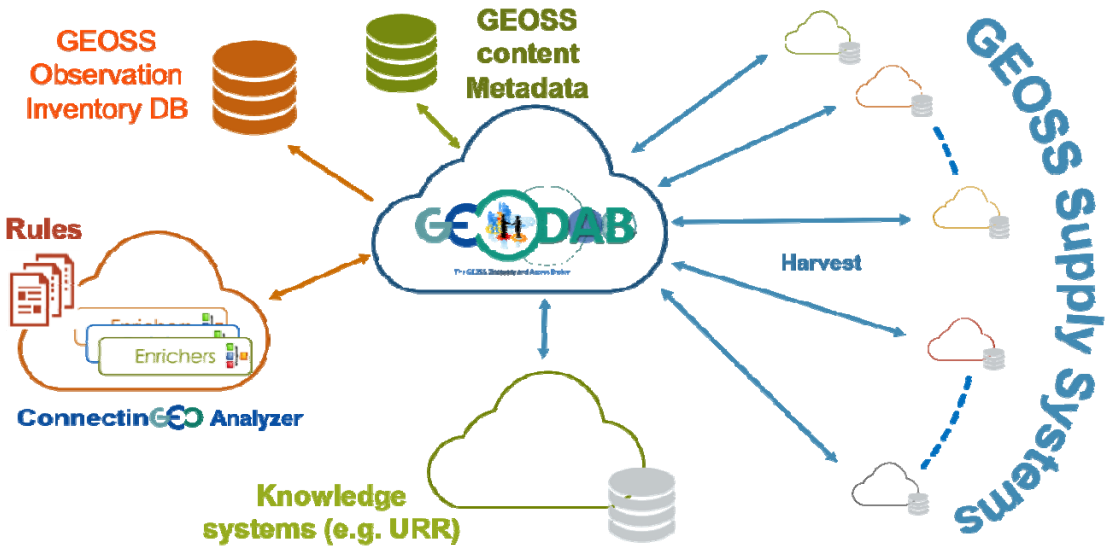

Figure 5 – High-Level Architecture of OI

GEOSS OI DB

A new No-SQL DB (the GEOSS OI DB) contains enriched metadata content of GEOSS OI and is based on the DB schema defined by ConnectinGEO D4.1.

The GEO DAB can directly read/write the GEOSS OI DB. Therefore, users can utilize the GEO DAB APIs to access the GEOSS OI DB. These APIs facilitate the development of web tools accessing the ConnectinGEO content to create plots, reports, summary statistics, and other useful inputs to the gap analysis.

ConnectinGEO Analyzer

The ConnectinGEO Analyzer is the component that implements the enrichment of the provided GEOSS metadata. It achieves the following tasks:

1. To read the actual GEOSS metadata content via the GEO DAB.
2. To enrich the metadata content accessing external knowledge bodies (see next section)
3. To write the enriched metadata to the GEOSS OI DB via the GEO DAB functionalities.

GEO DAB

The GEO DAB is in charge of:

1. To read the GEOSS metadata content
2. To provide read/write capabilities for the GEOSS OI DB
3. To provide a harmonized access to the external knowledge bodies needed for the metadata enrichment

In order to connect to external knowledge systems, GEO DAB functionalities must be extended to broker new types of sources (e.g. URR, scientific publication repositories, etc.).

3.1 ConnectinGEO Analyzer

The ConnectinGEO Analyzer is the core component to create the enriched content of GEOSS OI DB. Starting from a GEOSS metadata, the task of the ConnectinGEO Analyzer is to extract/infer extra information to enrich the GEOSS metadata. To this goal, the ConnectinGEO Analyzer makes use a set of Enrichers. Each Enricher is in charge of adding extra information of specific type (e.g. accessibility, data type, measurement, etc.).

Figure 6 depicts a simplified schema of the process implemented by the ConnectinGEO Analyzer.

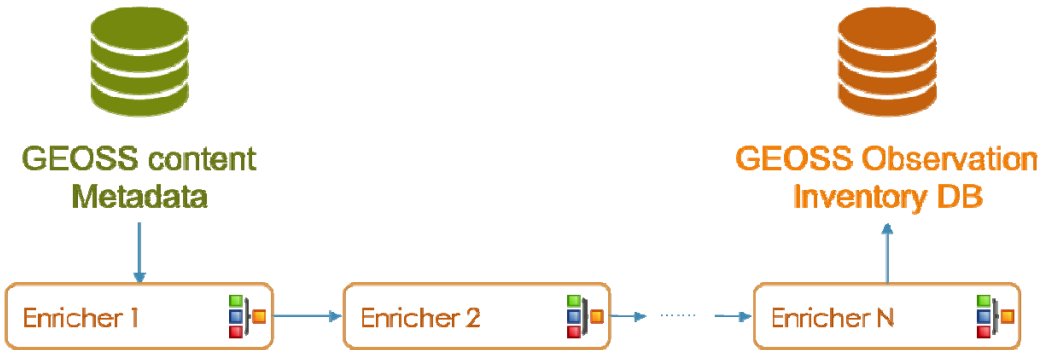

Figure 6 – ConnectinGEO Analyzer Process

GEOSS metadata content is passed to a pipeline of Enrichers. After the last Enricher completes its execution, the resulting enriched metadata is stored to the GEOSS OI DB.

The pipeline of Enrichers can be dynamically configured. This provides a high flexibility level, which is needed to adapt the enrichment process to the requirements emerging from the related tasks of the project.

3.2 Enricher Types

Current list of Enrichers and associated development status is shown in *Table* 1. More Enrichers might be needed in the second development loop addressing new requirements from related tasks in the project.

Table 1 – Current List of Enrichers

Enricher Type	Development Status
Web Resource Enricher	Done
Accessibility Enricher	Under Test
Record Type Enricher	Under Development
Data Enricher	Scheduled
Document Enricher	Scheduled

3.2.1 Web Resource Enricher

For each of the GEOSS Supply System, this Enricher is in charge of getting metadata records and creating an enriched version of it with the required indexed fields as defined by ConnectinGEO Deliverable 4.1.

The GEO DAB action required by the Enricher is the Harvesting of a GEOSS Supply System.

3.2.2 Accessibility Enricher

The Enricher is in charge of adding accessibility information of incoming metadata content.

The Record Enricher requires the GEO DAB to check if the referenced links (if any) in the metadata are accessible and adds this information to the metadata.

For each referenced link, the specific field that is added by the enricher is one of the ones listed in the table below.

Field Name	Field Value
accessibleLink	URL
unaccessibleLink	URL

3.2.3 Record Type Enricher

This Enricher is dedicated to add information about the type of data (e.g. Essential Variable, Indicator, Indexes, etc.) described by the incoming metadata content.

To do this, a set of tags is extracted from the metadata content. The tags are then used to determine the described data type based on information from external knowledge systems (e.g. URR) accessed by the GEO DAB.

The specific field that is added by the enricher is one of the ones listed in the table below.

Field Name	Field Value
EV	EV Name
Indicator	Indicator Name
Index	Index Name

3.2.4 Data Enricher

The task of the Data Enricher is to add information about the data (if accessible) referenced by the metadata.

The GEO DAB is invoked here to download the referenced data and provide related information (e.g. protocol used to access the data, format of the data, etc.).

For each referenced data that is downloadable, the specific fields added by this enricher are listed in the following Table.

Field Name	Field Value
ServiceProtocol	Protocol of the access service
DataFormat	Data format of the downloaded data
DownloadURL	The URL to download the data

3.2.5 Document Enricher

The task of the Document Enricher is to add information about the document (if accessible) referenced by the metadata.

The GEO DAB is invoked here to download the referenced document and provide related information (e.g. web page, scientific publication, etc.).

For each referenced document (if accessible), the specific field that is added by the enricher is one of the ones listed in the table below.

Field Name	Field Value
documentFormat	WebPage, PDF, etc.
documentType	ScientificPublication, TechnicalReport, etc.

3.3 MapReduce Framework of ConnectinGEO Analyzer

Due to the dimension of the GEOSS Metadata DB (millions of entries), Hadoop MapReduce[2] was selected as the framework to run the ConnectinGEO Analyzer.

The process described in section 2 is implemented as a MapReduce job running on a Hadoop cluster. Figure 7 depicts a simplified view of the MapReduce implementation.

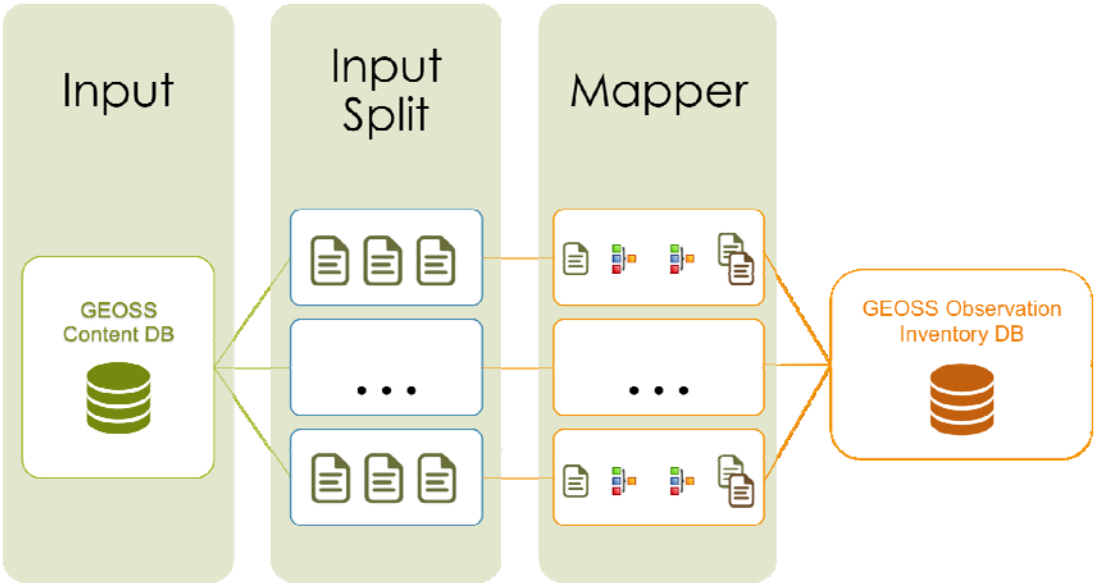

Figure 7 – Implementation of OI Population in MapReduce

In the MapReduce framework, the input of a job is split into a set of *InputSplits* and each *InputSplit* is processed by a *Mapper*. Each *InputSplit* is divided into records, and the *Mapper* processes each record. *InputSplit* is basically a list of records to be processed by one *Mapper*. This input-splitting process is used by the MapReduce framework to parallelize the execution of the job. In fact,

[2] http://hadoop.apache.org

after creating the *InputSplits*, these can be processed in parallel according to the availability of resources in the MapReduce cluster.

In the ConnectinGEO Analyzer use case, where the input of the job is the entire GEOSS metadata content, the framework implements an ad hoc splitter (which can be configured to generate a variable number *InputSplits*). This is in charge of reading a subset of GEOSS Metadata Content DB (step 2.1) and pass it to the *Mapper*.

A Connecting GEO Analyzer *Mapper* invokes a configurable list of Enrichers that work on a single metadata document (step 2.2). After all Enrichers have been executed the *Mapper* writes the output to the new DB (step 2.3).

Finally, it is worth to notice that presently no *Reducer* is used by the ConnectinGEO Analyzer framework. In fact, *Reducers* in MapReduce are used to aggregate/sort output of *Mappers*, which is not the case for the defined MapReduce job.

4 Accessing Observation Inventory

The developed OI is accessible by client applications using the GEO DAB supported interfaces[3].

In order to support the ConnectinGEO gap analysis, the OI content can be accessed by means of the GEO DAB JavaScript APIs[4]. Such APIs allow web applications to easily interrogate the OI. This way, web-based analysis tools can provide useful views and statistics of currently available observations from GEOSS can exploit the content of the OI.

Besides, an ad-hoc extension to the GEO DAB APIs was introduced in order to use JavaScript for submitting complex queries (i.e. with nested logical operators) to the OI. This is described in the next section.

4.1 Observation Inventory APIs

GEO DAB APIs allow combining different queryable fields with *AND* relation only; besides, it is not possible to create nested logical groups of constraints – e.g. *[keyword='temperature' AND (title='air' OR title='sea')]*.

To support gap analysis tools, such functionalities are very important since it might be needed to create complex queries taking into account observation requirements from the project related tasks.

[3] http://www.geodab.org

[4] http://api.eurogeoss-broker.eu

To support this requirement, an extension of GEO DAB APIs was developed. GEO DAB APIs have a defined extension point: the *key-value-pair (KVP)* parameter of the GEO DAB APIs *discover* method[5]. In addition to the documented values which can be passed using the *kvp* parameter, the OI supports the following *JSON*[6] encoding of complex queries:

```
jsonquery:{  "condition": "AND" | "condition": "OR",
                    "rules": [rule1, rule2,.., ruleN]}
```

where

```
rule:{  "condition": "AND" | "condition": "OR",
          "rules": [rule_1, rule_2,.., rule_N]}
```

or

```
rule: {"id": "QueryableID",
          "type": "QueryableTYPE ",
          "operator": "QueryableOPERATOR ",
          "value": "ConstraintValue"}
```

Thus the example query *[keyword='temperature' AND (title='air' OR title='sea')]* is encoded as in the following

```
jsonquery:{  "condition": "AND",
                    "rules": [{"id": "apiso:keyword",
                                  "type": "string",
                                  "input": "text",
                                  "operator": "contains",
                                  "value": "temperature"},
                                  {"condition": "OR",
                                    "rules": [{"id": " apiso:title",
                                                  "type": "string",
                                                  "operator": "contains",
```

[5] http://api.eurogeoss-broker.eu/docs/classes/DAB.html#method_discover

[6] http://json.org

```
                                    "value": "air"},
                                    {"id": " apiso:title",
                                      "type": "string",
                                      "operator": "contains",
                                      "value": "sea"}]
                            }]
                }
```


For each queryable field, *Table* 2 lists the *QueryableID*, *QueryableTYPE* and the list of supported *QueryableOPERATOR*s. This list might be subject to change in the second loop of development, an updated version will be published online on the Simple Web Client page (see next section).

Table 2 – Queryable Fields, Types and Supported Operators

Field	Queryable ID	Queryable Type	5 Supported Operators	6 Value Format
Abstract	apiso:abstract	string	contains	
Area	apiso:BoundingBox	string	is_contained, disjoint, overlaps	±nn.nn; ±ee.ee; ±ss.ss; ±ww.ww
Time Extent Begin	apiso:TempExtent_begin	date	less, less_or_equal, greater_or_equal, greater	YYYY-MM-DD
Time Extent End	apiso:TempExtent_end	date	less, less_or_equal, greater_or_equal, greater	YYYY-MM-DD
Essential Variable	essi:EVname	string	contains, equal	
Access Link	essi:HasAccessLinkage	string	equal	Yes | No

Other Link	essi:HasOtherLinkage	string	equal	Yes | No
Identifier	iso:identifier	string	equal	
Keyword	apiso:keyword	string	contains, equal	
Legal Access Constarints	essi:CNT.Leg.Access	string	contains, equal	*
Legal Other Constarints	essi:CNT.Leg.Other	string	contains, equal	*
Legal Use Constarints	essi:CNT.Leg.Use	string	contains, equal	*
Measurement	gvq:MeasureDescription	string	contains, equal	
Parent Identifier	apiso:ParentIdentifier	string	contains, equal	
Producer Name	apiso:Creator	string	contains, equal	
Published	apiso:PublicationDate	date	less, less_or_equal, greater_or_equal, greater	YYYY-MM-DD
Resolution	iso:GMI.BandResolution	double	less, less_or_equal, greater_or_equal, greater	
Theme	apiso:subject	string	contains, equal	
Title	apiso:title	string	contains, equal	

Topic Category	iso:TopicCategory	string	contains, equal
Use Limitation	essi:CNT.UseLimit.text	string	contains, equal

* These fields can be used both as Boolean and as Text matching fields. The format is

type;;*value*

Where type is one of true, false or text. When type is text, the field works as a Text matcher, and all records containing the string in *value* are accepted. Below is a rule example for this case, all records containing the text "copyright" in the Access constraint field will match:

{"id": "essi:CNT.Leg.Access", "type": "string", "operator": "contains", "value": "text;; copyright"}

Otherwise, the field works as a Boolean field and all records containing (if *type*=true) or not containing (if *type*=false) the corresponding metadata element are accepted. Below is a rule example for this case, all records containing the any text in the Access constraint field will match:

{"id": "essi:CNT.Leg.Access", "type": "string", "operator": "contains", "value": "true;;undefined"}

7.1 Observation Inventory Simple Web Client

A simple Web Client was developed to interrogate the OI. The objective of this web client is to allow a basic exploration of the OI content. It is possible to query the OI using the available queryable fields (as defined by ConnectinGEO Deliverable 4.1). The Web Client makes use of the above-described extension of the GEO DAP APIs to submit complex queries to the OI.

Figure 8 depicts a screenshot of current Web Client showing the results of a query.

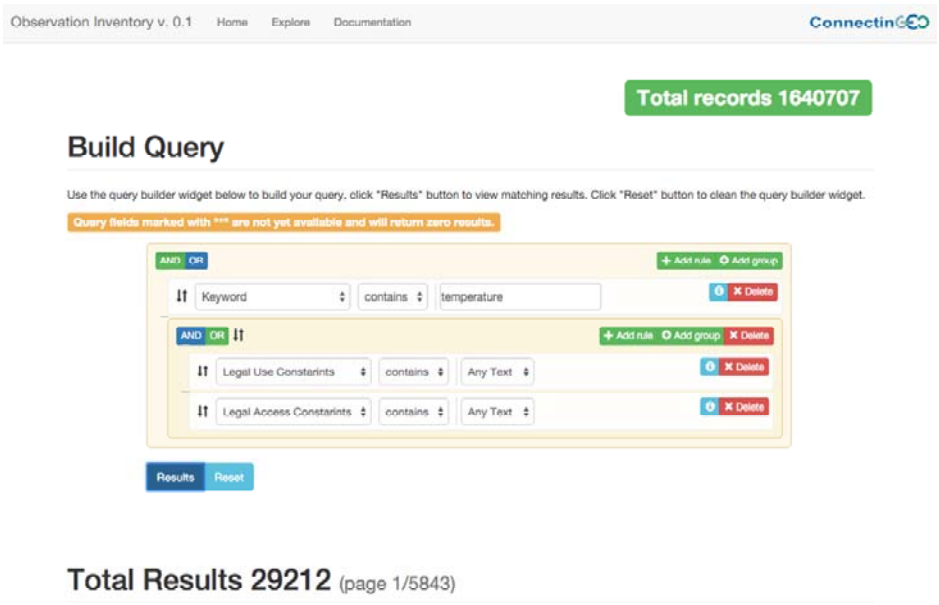

Figure 8 – Simple Web Client of OI

The Simple Web Client is available at

http://oi.geodab.eu/oi-client/home/

8 Conclusions

WP4 defined a high-level process for the population of the Observation Inventory (OI). Based on this process, the architecture for the creation of the OI was designed and developed.

The first version of the OI was created and populated using the current information in the metadata concentrated in the GEO Discovery and Access Broker (DAB) of the GEOSS Common Infrastructure (GCI). The first population process was run in December 2015, resulting in a total of more than 1.6M harvested metadata records.

The developed OI is accessible online by client applications using all service interfaces supported by the GEO DAB, including JavaScript APIs with an ad hoc extension to execute complex queries. This way, it is possible to use the OI as a data source for different analysis tools, which create plots, reports, or summary statistics useful for the ConnectinGEO gap analysis.

A simple Web Client was developed to demonstrate how to interrogate the OI. In the “Documentation” section, the Simple Web Client provides also basic examples of how the developed OI can be used by web-based tools to provide views and statistics of currently available observations useful for the gap analysis.

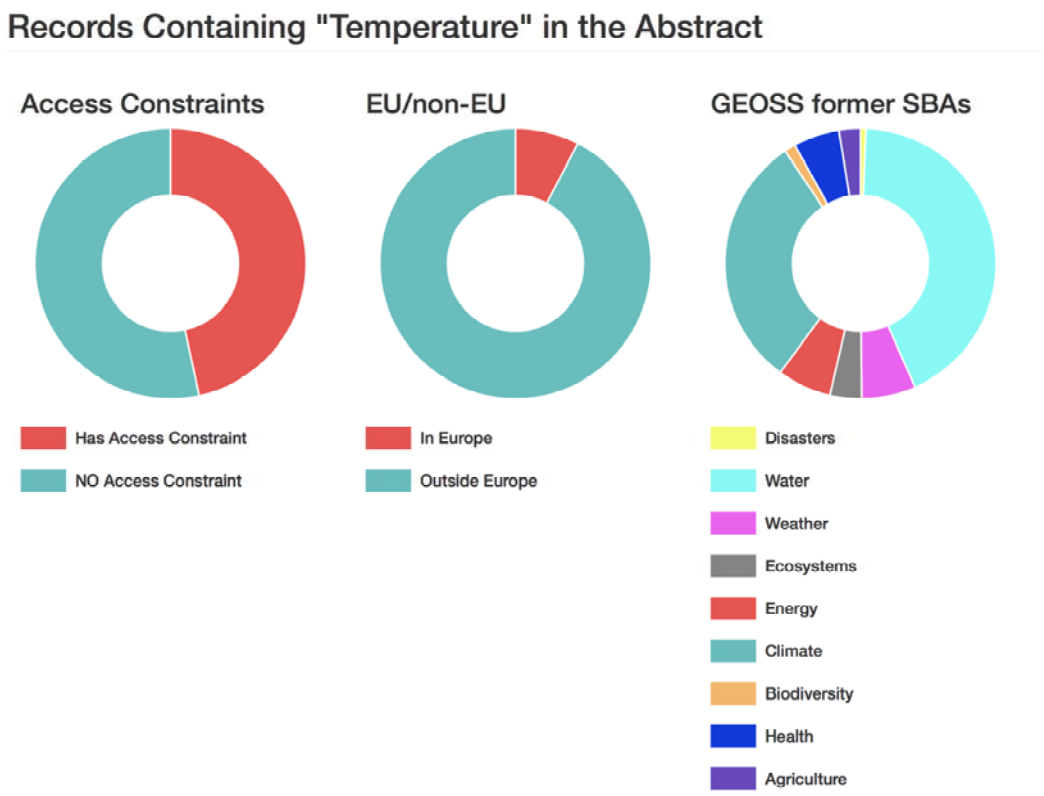

Figure 9 – Simple Web Client Example

Figure 9 depicts one of these examples. Taking into account all records containing “temperature” in the metadata abstract field, three charts are created: the first showing which ones have access constraints, the second one showing which records cover a spatial extent inside Europe, and the third one their distribution across GEOSS former SBAs.

Finally, the developed framework is ready to be extended and complement current information with data extracted from additional external sources (URR 2.0, scientific literature DBs, etc.) and the results of ConnectinGEO Task 2.3.